SPECIAL VERSUS STANDARD ALGORITHMS FOR LARGE-SCALE HARVEST SCHEDULING PROBLEMS

by

Chiun-Ming Liu

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Industrial Engineering and Operations Research

APPROVED:

[illegible]

Hanif D. Sherali, Chairman

[illegible]

Subhash C. Sarin

[illegible]

Ching-Chung Wang

December, 1988

Blacksburg, Virginia

SPECIAL VERSUS STANDARD ALGORITHMS FOR LARGE-SCALE HARVEST SCHEDULING PROBLEMS

by

Chiun-Ming Liu

Hanif D. Sherali, Chairman

Industrial Engineering and Operations Research

(ABSTRACT)

This thesis is concerned with structure exploitation and the design of algorithms for solving large-scale harvest scheduling problems. We discover that the harvest scheduling problem involving area constraints possesses a network structure. In Model I-Form 1, the network constraints form a separable block diagonal structure, which permits one to solve for the decision variables belonging to each individual area constraints as independent knapsack problems. In Model II-Form 1, the network constraints constitute a longest path problem, and a Longest Path Algorithm is developed to solve this problem in closed form. The computational time for this scheme is greatly reduced over that for the revised simplex method. The Dantzig-Wolfe algorithm is coded and tuned to solve general Model II problems, taking advantage of the Longest Path Algorithm in the subproblem step, and using the revised simplex method to solve the master problems. Computational results show that the algorithm solves problems to within one percent accuracy far more efficiently than the revised simplex method using MPS III. Both the CPU time and number of iterations for the Dantzig-Wolfe algorithm are less than those for the MPS III, depending on the problem size. Results also suggest that the Dantzig-Wolfe algorithm makes rapid advances in the initial iterations, but has a slow convergence rate in the final iterations. A Primal-Dual Conjugate Subgradient Algorithm is also coded and tuned to solve general Model II problems. Results show that the computational effort is greatly affected by the number of side constraints. If the number of side constraints is restricted, the Primal-Dual Conjugate Subgradient Algorithm can give a more efficient algorithm for solving harvest scheduling problems. Overall, from a storage requirement viewpoint, the Primal-Dual Conjugate Subgradient Algorithm is best, followed by the Dantzig-Wolfe algorithm and then the revised simplex method. From a computational efficiency viewpoint, if the optimality criterion is suitably selected, the Dantzig-Wolfe algorithm is best, provided that the number of side constraints are not too many, followed by the revised simplex method and then the Primal-Dual Conjugate Subgradient Algorithm.

Acknowledgements

I wish to thank the members of my committee, Drs. Subhash C. Sarin and Ching-Cheng Wang, and Dr. G. V. Loganathan, for their interest and help. A special note of appreciation must go to my committee chairman, Dr. Hanif D. Sherali. He has been my adviser and friend, and offered valuable help during this study. I also want to thank Dr. Harold E. Burkhart for providing a grant to support this study. Finally, I need to express my full appreciation to my wife, and our children, and for their support and patience.

Table of Contents

List of Illustrations

List of Tables

Chapter I

INTRODUCTION

Timber harvest schedules form the heart of forest management and timber supply. They prepare harvest activity information of when, where, how, and how much timber should be cut and grown to reach owner objectives. The only scheduling models available up to the 1960's were simple formulas and simulations, augmented by map overlaps and human experience. These approaches and analytical methods proved inadequate for the practical scheduling tasks facing the USDA Forest Service and the forest industry.

With the advent of the computer, full-scale forest management planning finally became possible, and one can now incorporate aspects of best management practices and alternative treatments. The development of linear programming models not only permits the introduction of a wide range of management practices characteristic of intensive management, but also opens up opportunities of analysis in broaden aspects of the evolutionary transition from functional to integrated land management planning.

Linear programming is a mathematical technique that allows decision makers to compare the ability of alternative management strategies to meet stated goals within available resource limitations. During the past decades, this technique has been used to optimize timber harvest and investment on public and private forests. Computerized LP models have thus become popular not because the models make decisions, but because they can facilitate better decisions by evaluating several hundred thousand decision variables in a fraction of the time it would take to locate the best combination of decision variables by hand.

The simplex method has been used to solve LP models. Very often, the formulation of timber harvest scheduling problems is large and requires large storage capacity. It is quite expensive to perform numerous applications of these models to examine a large number of management alternatives using the simplex method. For large-scale problems, the simplex method might not be the best algorithm. So, the need for a more efficient solution algorithm for solving large-scale harvest scheduling problems becomes appealing.

Fortunately, many harvest scheduling problems almost always have special structure. By developing specialized solution algorithms to take advantage of this structure, significant gains in computational efficiency can be achieved. During the past decades, many classes of structured problems have been identified and a great many algorithms have been developed for their solution.

In forest management, very little effort has been made to exploit possible special structure in timber harvest scheduling problems, and to develop specialized solution algorithms to take advantage of this structure. Once this approach is proved efficient, the exploitation of special structure and the development of specialized solution algorithms might be alternatives to the simplex method or any other computerized solution techniques.

OBJECTIVES

The objectives of this study are two-fold. First, to identify the structure of general timber harvest scheduling models. Second, to develop some specialized solution algorithms for solving these models.

Chapter II

FORMULATION OF HARVEST SCHEDULING PROBLEMS

The forest-wide decision for harvest scheduling problems is how much timber to cut and when. Decision variables are structured to respond to the question: How many acres should be cut from each cutting unit in each period? A cutting unit is defined as a collection of acres from across the forest which share similar biological, silvicultural, and economic attributes. There are several methods for creating cutting units. These at least include: (1) each currently existing stand can constitute a cutting unit; and (2) cutting units can be created by consolidating geographically contiguous stands which, after the initial harvest, are managed as one stand.

The development of a harvest schedule involves the assignment of a management regime to each cutting unit. Each management regime defines a strategy involving a series of harvest/silvicultural practices that can be implemented during the planning period. So, a series of harvest activities should be constructed, each activity representing an alternative sequence of clearcutting, commercial thinning, and/or precommercial thinning, spanning many decades to the planning horizon.

Clearcutting activity is the major harvest practice and has been considered in most forest management studies (e.g. Clutter et al. 1983). Only clearcutting activity is considered in this study. If commercial thinning or precommercial thinning activity is considered, the number of possible management regime will increase. Associated with each type of clearcutting activity are decision variables, which keep track of acreage within cutting units. Generally, more constraints are added to the model in addition to those needed to track acreage within cutting units. These constraints control the rate of change in timber volume harvested and budget available from period to period. The objective function used is to maximize net present value or physical product, or to minimize the production cost, according to the nature of the problems.

Johnson and Scheurman (1977) outline the generic structure of harvest scheduling problems, labeling them "Model I" and "Model II". The primary difference in Model I and Model II is in the aggregation of cutting units through time. Model I maintains cutting unit identity and does not allow cutting unit merging. Model II can allow cutting units of like characteristics to be merged together into one cutting unit, regardless of geographic location, cutting unit origin, etc. Therefore, the formulation of harvest scheduling problems is classified as Model I and Model II, according to the definition of decision variables. In addition to area constraints, even-flow constraints and non-declining harvest volume and cash flow constraints are respectively considered in this study. Even-flow constraints restrict the periodic harvest volume at the same level over the planning horizon. Consequently, the objective function is to maximize this amount of even-flow or to maximize the minimum volume harvested over the periods. Non-declining harvest volume constraints imply that the periodic harvest volume should be non-decreasing to meet the general trend of stumpage demand.

MODEL I

In this approach, each cutting unit in the first period retains its integrity throughout the planning period. Each harvest activity represents a possible management regime for a particular cutting unit (Table 1). So the decision variable is defined as the area of one cutting unit assigned to some management regime. The objective function of this model is to maximize net present value or timber harvest volume. The area constraints require the total acres assigned to the different management regimes for each cutting unit to equal the specified cutting unit acreage. First, only area constraints are considered. Then even-flow constraints are added to the area constraints. Finally, general constraints, including non-declining constraints, harvest area constraints, and/or cash flow constraints, are considered and added to the models.

The variables used to define the Model I problem are as follows.

n = the number of cutting units
m = the number of management regimes
p = the number of cutting periods
h = the even flow in each cutting period
x_{ik} = the acres of cutting unit i assigned to management regime k
c_{ik} = the contribution to the objective function per acre for decision variables x_{ik}
b_i = the acres contained in cutting unit i
v_{ijk} = the per-acre volume harvested from cutting unit i in period j if management regime k is used
r_{ijk} = the per-acre cash flow from cutting unit i in period j if management regime k is used

The formulation of Model I including area constraints is given below.

Model I-Form 1:

$$\text{Maximize} \quad \sum_{i=1}^{n}\sum_{k=1}^{m} c_{ik}x_{ik}$$

$$\text{Subject to} \quad \sum_{k=1}^{m} x_{ik} = b_i \qquad i = 1, \dots, n$$

Table 1. All possible management regimes of Model I with one cutting unit and six cutting periods.

Management regime	Period number					
	1	2	3	4	5	6
1	C					
2	C		C			
3	C		C		C	
4	C		C			C
5	C			C		
6	C			C		C
7	C				C	
8	C					C
9		C				
10		C		C		
11		C		C		C
12		C			C	
13		C				C
14			C			
15			C		C	
16			C			C
17				C		
18				C		C
19					C	
20						C
21						

Note: C represents a clearcutting activity.

$$x_{ik} \geq 0 \qquad i = 1, \dots, n \quad k = 1, \dots, m$$

Assuming that there is no constraint on the maximum age a cutting unit can achieve before harvest, and all existing cutting units are of merchantable age, Johnson and Scheurman (1977) indicate that the number of decision variables can be estimated from the following formula:

$$nm = n(\sum_{i=1}^{p} m_i + 1)$$

where n = the number of cutting units
p = the number of cutting periods
m = the number of management regimes
m_i = the number of possible management regimes with i periods left

When the number of cutting periods left is less than or equal to the minimum periods between regeneration harvests, m_i equals one. When the number of cutting periods left is greater than the minimum periods, $m_i = m_{i-1} + m_{i-L}$, where L is the minimum periods between regeneration harvests.

This formula suggests that the number of decision variables depends on the number of cutting units, the minimum periods between regeneration harvests, and the number of cutting periods. Table 2 shows that the number of possible management regimes is increasing in a Fibonacci series when the number of periods left increases. For example, a harvest scheduling problem with 20 cutting periods, 2 minimum periods between regeneration harvests, and 15 cutting units might have 265,665 decision variables and 15 area constraints.

The USDA Forest Service has been using a procedure called "area-volume check" to calculate the maximum sustainable harvest volume from existing cutting units over a planning horizon. This procedure is equivalent to maximizing even-flow subject to area constraints. So the area-volume check approach can be formulated as follows.

Table 2. Series of possible management regimes with 6 planning periods and 2 minimum periods between regeneration harvests.

Period when first harvest occurs	Number of periods left	Number of possible management regimes
6	1	1
5	2	1
4	3	2
3	4	3
2	5	5
1	6	8

Model I-Form 2:

$$\text{Maximize} \quad h$$

$$\text{Subject to} \quad \sum_{k=1}^{m} x_{ik} = b_i \quad i = 1, \dots, n$$

$$\sum_{i=1}^{n}\sum_{k=1}^{m} v_{ijk} x_{ik} \geq h \quad j = 1, \dots, p$$

$$x_{ik} \geq 0 \quad i = 1, \dots, n \quad k = 1, \dots, m$$

The formulation of maximizing even-flow subject to area constraints might use equalities for all of the volume constraints. However, it makes more sense to solve the lesser restricted problem of maximizing the minimum volume harvested over the periods. So, inequalities instead of equalities are used here for the volume constraints. Further, we may wish to impose more constraints in addition to the area constraints, including non-declining harvest volume constraints, harvest area constraints, and/or cash flow constraints. So the formulation of general Model I problems is given as follows.

Model I-Form 3:

$$\text{Maximize} \quad \sum_{i=1}^{n}\sum_{k=1}^{m} c_{ik} x_{ik}$$

$$\text{Subject to} \quad \sum_{k=1}^{m} x_{ik} = b_i \quad i = 1, \dots, n$$

$$\sum_{i=1}^{n}\sum_{k=1}^{m} v_{ijk} x_{ik} \leq \sum_{i=1}^{n}\sum_{k=1}^{m} v_{i\,j+1\,k} x_{ik} \quad j = 1, \dots, p-1$$

$$\sum_{i=1}^{n}\sum_{k=1}^{m} r_{ijk} x_{ik} \leq \sum_{i=1}^{n}\sum_{k=1}^{m} r_{i\,j+1\,k} x_{ik} \quad j = 1, \dots, p-1$$

$$x_{ik} \geq 0 \quad i = 1, \dots, n \quad k = 1, \dots, m$$

Inclusion of these restrictions would add more constraints, but not decision variables, to the model. Each of these restrictions increase the number of constraints by roughly the number of cutting periods. Also, the inclusion of harvest volume and cash flow constraints complicates the model structure. In Model I-Form 1 where only area constraints are imposed, these constraints form a separable block diagonal structure, which permits one to solve the problem as a series of knapsack problems. In Model I-Form 2 or 3, the volume constraints and/or cash flow constraints form complicating side constraints which increase the solution difficulty of these problems.

MODEL II

In Model II, each stand containing area in the first period forms a cutting unit until this area is harvested and regenerated. When an area is harvested, it denotes the act of removing the trees on this area. Regeneration means the act of replacing old trees removed with others of a new generation. All area that is harvested and regenerated in each period forms a new cutting unit until it is regeneration harvested. Regeneration harvestings are made with the two purposes of removing the old trees and creating environmental conditions favorable for establishment of reproduction. In general, we need four types of decision variables: (i) area of initial cutting unit i that is harvested in cutting period j; (ii) area that is regenerated in cutting period j and harvested in cutting period k; (iii) area of initial cutting unit i that is left uncut through the planning period; and (iv) area regenerated in cutting period j that is left uncut through the planning period. Since the initial cutting units do not retain their identity through the planning period, the initial cutting units are referred to as the initial strata according to stand age, species, and growth conditions in existence at the beginning of the planning period.

The objective function is to maximize net present value or timber harvest volume. The calculation of net present value or timber harvest volume includes the ending inventory in Model II formulation but not in Model I. Two types of area constraints are needed to force the total acres assigned to any cutting unit to equal the specified cutting unit acreage. The variables used in the formulation of Model II are defined as follows.

m = the number of initial strata
p = the number of cutting periods
L = the minimum periods between regeneration harvests
y_{ij} = the acres of initial stratum i that are harvested in cutting period j
x_{jk} = the acres regenerated in cutting period j and harvested in cutting period k
c_{ij} = the contribution to the objective function per acre for decision variables y_{ij}
d_{jk} = the contribution to the objective function per acre for decision variables x_{jk}
w_i = the acres of initial stratum i that are left uncut through the planning period
q_j = the acres regenerated in cutting period j and left uncut through the planning period
f_i = the contribution to the objective function per acre for decision variables w_i
g_j = the contribution to the objective function per acre for decision variables q_j
u_{ij} = the volume per acre harvested in cutting period j from stands belonging to initial stratum i
v_{jk} = the volume per acre harvested in period k from stands regenerated in period j
r_{ij} = the per-acre cash flow in cutting period j from stands belonging to initial stratum i
s_{jk} = the per-acre cash flow in period k from stands regenerated in period j

After harvest, we assume that the cutting units are converted into stands with the same species and growth conditions. This assumption can be relaxed by expanding the dimension of each variable. The formulation of Model II including area constraints is given below.

Model II-Form 1:

$$\text{Maximize} \quad \sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j$$

$$\text{Subject to} \quad \sum_{j=1}^{p} y_{ij} + w_i = b_i \quad i = 1, \dots, m$$

$$\sum_{k=j+L}^{p} x_{jk} + q_j = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij} \quad j = 1, \ldots, p$$

$$y_{ij} \geq 0 \quad i = 1, \ldots, m \quad j = 1, \ldots, p$$

$$x_{jk} \geq 0 \quad j = 1, \ldots, p - L \quad k = j + L, \ldots, p$$

$$w_i \geq 0 \quad i = 1, \ldots, m$$

$$q_j \geq 0 \quad j = 1, \ldots, p$$

Assuming that there is no constraint on the minimum age a cutting unit can achieve before harvest, and that all existing cuttings are of merchantable age, Johnson and Scheurman (1977) indicate that the number of decision variables for Model II is:

$$mp + \frac{(p - L + 1)(p - L)}{2} + (m + p).$$

where m = the number of initial strata
p = the number of cutting periods
L = the minimum periods between regeneration harvests

The first term in this formula equals the number of possible harvest activities from all the possible initial strata, since for each initial stratum, there are p possible harvest activities. The second term equals the number of possible harvest activities from all the possible new cutting units that can be harvested again during the planning horizon, since each new cutting unit has $(p - L - i + 1)$ possible harvest activities if this cutting unit is created in the ith period. The last term equals the number of possible harvest activities from the uncut units that exist at the start of the planning horizon and are created during the planning horizon. Usually, the formulation of Model II needs less decision variables and more area constraints than Model I. For example, a harvest scheduling problem with 20 cutting periods, 2 minimum periods between regeneration harvests, and 15 cutting units requires 506 decision variables and 35 area constraints.

Further, we may want to add even-flow harvest constraints to restrict the fluctuation in harvest levels from period to period. So the formulation of Model II including even-flow constraints is given as follows.

Model II-Form 2:

Maximize h

Subject to $\sum_{j=1}^{p} y_{ij} + w_i = b_i \quad i = 1, \dots, m$

$$\sum_{k=j+L}^{p} x_{jk} + q_j = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij} \quad j = 1, \dots, p$$

$$\sum_{i=1}^{m} u_{ij} y_{ij} + \sum_{k=1}^{j-L} v_{kj} x_{kj} \geq h \quad j = 1, \dots, p$$

$$y_{ij} \geq 0 \quad i = 1, \dots, m \quad j = 1, \dots, p$$
$$x_{jk} \geq 0 \quad j = 1, \dots, p - L \quad k = j + L, \dots, p$$
$$w_i \geq 0 \quad i = 1, \dots, m$$
$$q_j \geq 0 \quad j = 1, \dots, p$$

Inequalities instead of equalities are used for the volume constraints for the reason mentioned above. In the formulation of general harvest scheduling problems, we may impose restrictions like harvest volume constraints, harvest area constraints, and/or cash flow constraints. In this case, inclusion of these restrictions would increase the number of constraints, but add no more decision variables, to the model. However, the model structure becomes more complicated with additional side constraints, since as we show later, the area constraints possess a network structure. The formulation of general Model II problems is given below.

Model II-Form 3:

Maximize $\sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j$

Subject to $\sum_{j=1}^{p} y_{ij} + w_i = b_i \quad i = 1, \dots, m$

$$\sum_{k=j+L}^{p} x_{jk} + q_j = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij} \quad j = 1, \dots, p$$

$$\sum_{i=1}^{m} u_{ij}y_{ij} + \sum_{k=1}^{j-L} v_{kj}x_{kj} \leq \sum_{i=1}^{m} u_{i\,j+1}y_{i\,j+1} + \sum_{k=1}^{j+1-L} v_{k\,j+1}x_{k\,j+1} \quad j = 1, \dots, p-1$$

$$\sum_{i=1}^{m} r_{ij}y_{ij} + \sum_{k=1}^{j-L} s_{kj}x_{kj} \leq \sum_{i=1}^{m} r_{i\,j+1}y_{i\,j+1} + \sum_{k=1}^{j+1-L} s_{k\,j+1}x_{k\,j+1} \quad j = 1, \dots, p-1$$

$$y_{ij} \geq 0 \quad i = 1, \dots, m \quad j = 1, \dots, p$$

$$x_{jk} \geq 0 \quad j = 1, \dots, p-L \quad k = j+L, \dots, p$$

$$w_i \geq 0 \quad i = 1, \dots, m$$

$$q_j \geq 0 \quad j = 1, \dots, p$$

Chapter III

LITERATURE REVIEW

Forest management planning and timber supply have been studied by operations research analysts for quite some time. Most recent developments of harvest scheduling problems can be found in Bare et al. (1984) and Iverson and Alston (1986). A majority of these studies concerning timber harvest scheduling problems have relied heavily on linear programming models.

Two different model formulations have been adopted for outlining the general structure of the harvest scheduling problems. Both problem types reflect common ways to view the scheduling problem in industrial and public forest management. Johnson and Scheurman (1977) outline these generic structures, labeling them "Model I" and "Model II". The key to understanding the distinction between them is found in the definition of decision variables. In Model I, decision variables trace harvest activities from the existing stand through several harvests on future regenerated stands. The Timber Resource Allocation Model (Timber RAM) developed by the USDA Forest Service (Navon 1971) and MAX-MILLION developed at the University of Georgia in collaboration with a number of industrial cooperations (Clutter 1968, Ware and Clutter 1971) represent the application of linear programming to solve the Model I formulation.

In contrast, Model II is typified by a separation of decision variables. One set of decision variables traces harvest activities on the existing cutting units. A separate set of decision variables traces actions on the timberland during the conversion of the subsequent cutting units. Each time a cutting unit is reestablished, it is traced with a new set of decision variables. Constraints link the two sets of decision variables and ensure that acres are properly tracked in an accounting sense. The Multiple Use Sustained Yield Calculation (MUSYC) developed by Johnson et al (1979), and the Forest Planning Model (FORPLAN) version 1 (Johnson et al. 1986a) and version 2 (Johnson et al. 1986b) belong to this type of formulation.

Johnson (1977) points out that the problem size for Model I is especially sensitive to minimum periods between regeneration harvests, and that the problem size for Model II is especially sensitive to the number of acreage grouping at each cutting period that must be maintained for future cutting units within each species-site. Which model is more computationally efficient in a given problem depends, among other things, on the user's preference relative to these two key parameters. In most cases, the Model I formulation has the advantage of preserving the location of the cutting units. This greatly facilitates translating the LP solution into operational plans and eventually into management decisions. But some harvest scheduling problems can be handled with greater computational efficiency through a Model II formulation.

The application of mathematical programming to harvest scheduling problems has surfaced problems related to solving large-scale linear programming models, a process that has proved to be costly using the simplex method. Over the past decades, very little effort has been directed in developing alternative techniques for solving large-scale harvest scheduling problems. One initial effort was made by Liittschwager and Tcheng (1967). They formulated and solved a forest scheduling problem using a linear programming decomposition method. Results showed that their decomposition approach gained little over the simplex method because a very large number of columns might induce the occurrence of dual degeneracy. The objective function value increases rapidly during early iterates, but convergence becomes slower as the computations progress.

Recently, Berck and Bible (1984) used the Dantzig-Wolfe (1960) decomposition method to solve a large-scale Model II-Form 2 harvest scheduling problem. They decomposed the original problem into a master problem with many subproblems according to the number of species, for each of which they had area constraints. The subproblems were solved using a simple, recursive routine. Results show that this price-directive decomposition technique reduces the computational burden in terms of time and computer storage, as compared with the simplex method.

Hoganson and Rose (1984) develop a simulation approach similar to the dual decomposition method to solving a Model I problem. In their approach, the primal problem is to minimize production cost subject to area constraints and production output level constraints. The corresponding dual problem is solved assuming that the value of dual variables associated with the output level constraints is known. The major steps of their approach are as follows.

Step 1. Estimate the dual variable values associated with the output level constraints.

Step 2. Solve the dual problem for the dual variable values associated with the area constraints.

Step 3. Solve the primal problem by using the complementary slackness conditions.

Step 4. Test for primal feasibility by checking the output level constraints.

Step 5. Update the dual variable values associated with the output level constraints if primal feasibility is not satisfied, and return to Step 2.

In Step 5, three procedures are developed to adjust the dual variable values or the marginal cost associated with each output level constraints. These three procedures are float, shape, and smooth procedures. The float procedure moves the entire marginal cost curve up or down, the shape procedure changes the shape of the marginal cost curve, and the smooth procedure adjusts the slope of the marginal cost curve. Results show that this approach uses only 3,224 dual variables instead of 43 thousand decision variables in solving one large-scale problem. The storage requirement is less and the computational time is reduced dramatically.

Theoretically, Berck and Bible's method and Hoganson and Rose's method are primal-dual strategies in that a dual solution is estimated and then a primal solution is derived from the given dual solution. The difference between Berck and Bible's method and Hoganson and Rose's method lies in the way the dual solutions are generated. In Berck and Bible's method, the original Model II-Form 2 problem is decomposed into a master problem and many subproblems. For each subproblem, a new variable is defined and the dual solution is obtained using the Karush-Kuhn-Tucker conditions. For the master problem, the shadow prices are derived and are used to update the coefficients of the objective function for each subproblem. In Hoganson and Rose's method, which is applied to a minimization type of the Model I-Form 2 problem, a group of key dual variables are assumed to be known and the remaining dual variables are solved for in terms of these key dual variables. Then some simulation steps are proposed to adjust the values of these key dual variables in order to determine improving solutions. The solution obtained is dual-feasible but primal-infeasible. Also, the solution depends on the adjustment steps used. Hoganson and Rose's method is applied only to a minimization type of the Model I problem. The results from Berck and Bible's and Hoganson and Rose's research reveal that the decomposition strategy might provide an efficient algorithm for solving general large-scale harvest scheduling problems.

Sen and Sherali (1986) develop a primal-dual subgradient algorithm for preferably decomposable, generally nondifferentiable, convex programming problems. This algorithm employs a Lagrangian dual function with a suitable penalty function to generate a sequence of primal and dual iterates. The steps for this generic algorithm are as follows.

Step 1. Select an initial pair of primal and dual iterates.

Step 2. Compute the Lagrangian dual and penalty functions at the given pair of primal and dual iterates.

Step 3. If the values from the Lagrangian and penalty functions are identical, terminate.

Step 4. Compute subgradients of the Lagrangian and penalty functions at the given pair of primal and dual iterate.

Step 5. Check the optimality conditions for either the primal or dual iterate.

Step 6. Compute the step length.

Step 7. Update the primal and dual iterates, and return to Step 2.

In this algorithm, the initial pair of primal and dual iterates are arbitrarily chosen. If we can carefully choose some initial pair of primal and dual iterates, the number of iterates to reach optimality might be reduced. Also, the step length choice is important for improving the computational efficiency, and depends on the selection of the penalty function and its parameters. This algorithm can be used to efficiently solve the quadratic objective function of Model I and II problems when the timber price is a function of the quantity offered (Johnson and Scheurman 1977).

Lasdon (1970) suggests that large-scale problems almost always have special structures. In mathematical programming, size is determined by the number of variables, the number and complexity of constraints, and the complexity of the objective function. What "large" means depends on the capabilities of overall solution algorithms. Fortunately, many harvest scheduling LP models can be manipulated to display blocks along the main diagonal in addition to a few coupling constraints. Exploitation of such special structures via the development of specialized solution algorithms makes it possible to improve the computational efficiency for solving general harvest scheduling models.

Chapter IV

STRUCTURE EXPLOITATION

The harvest scheduling models presented above have a large number of decision variables but few constraints. The area constraints appearing in either Model I or Model II exhibit special structure. In Model I, the area constraints form a network flow structure or a separable block diagonal structure, which permits one to solve for the decision variables belonging to each individual area constraint independently as a knapsack problem. Hence, if only area constraints appear in the formulation of Model I, this problem can be solved by separately maximizing over individual cutting units.

In Model II, we exhibit below that the area constraints possess a network flow structure. In this case, the nodes represent the initial cutting units and the periods, and the arcs represent cutting activities. Berck and Bible (1984) do not recognize this structure but they do prescribe an efficient solution procedure for this problem, using the Karush-Kuhn-Tucker (KKT) optimality conditions. The network formulation of the Model II problem becomes evident by adding a new constraint that is the negative of the sum of the other constraints. This formulation of Model II is given below.

NFP: Maximize $\sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j$

Subject to $\sum_{j=1}^{p} y_{ij} + w_i = b_i \quad i = 1, \dots, m$

$$\sum_{k=j+L}^{p} x_{jk} + q_j = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij} \quad j = 1, \dots, p$$

$$-\sum_{i=1}^{m} w_i - \sum_{j=1}^{p} q_j = -\sum_{i=1}^{m} b_i$$

$$y_{ij} \geq 0 \quad i = 1, \dots, m \quad j = 1, \dots, p$$
$$x_{jk} \geq 0 \quad j = 1, \dots, p-L \quad k = j+L, \dots, p$$
$$w_i \geq 0 \quad i = 1, \dots, m$$
$$q_j \geq 0 \quad j = 1, \dots, p$$

Consider the constraint matrix associated with NFP. Each row of this matrix corresponds to a node of the network, and each column to an arc. Each column of this matrix contains exactly two nonzero coefficients: a "+ 1" and a "-1". The column associated with arc (i,j) contains a "+ 1" in row i, a "-1" in row j, and zeros elsewhere. This constraint matrix is called the node-arc incidence matrix for the graph. The graph for this network is shown in Figure 1.

The solution to this network formulation becomes evident when one examines the corresponding dual problem to the original Model II problem without the redundant constraint. This dual is given as follows.

NFD: Minimize $\sum_{i=1}^{m} b_i \phi_i$

Subject to $\phi_i \geq c_{ij} + \gamma_j \quad i = 1, \dots, m \quad j = 1, \dots, p$

$$\gamma_j \geq d_{jk} + \gamma_k \quad j = 1, \dots, p-L \quad k = j+L, \dots, p$$

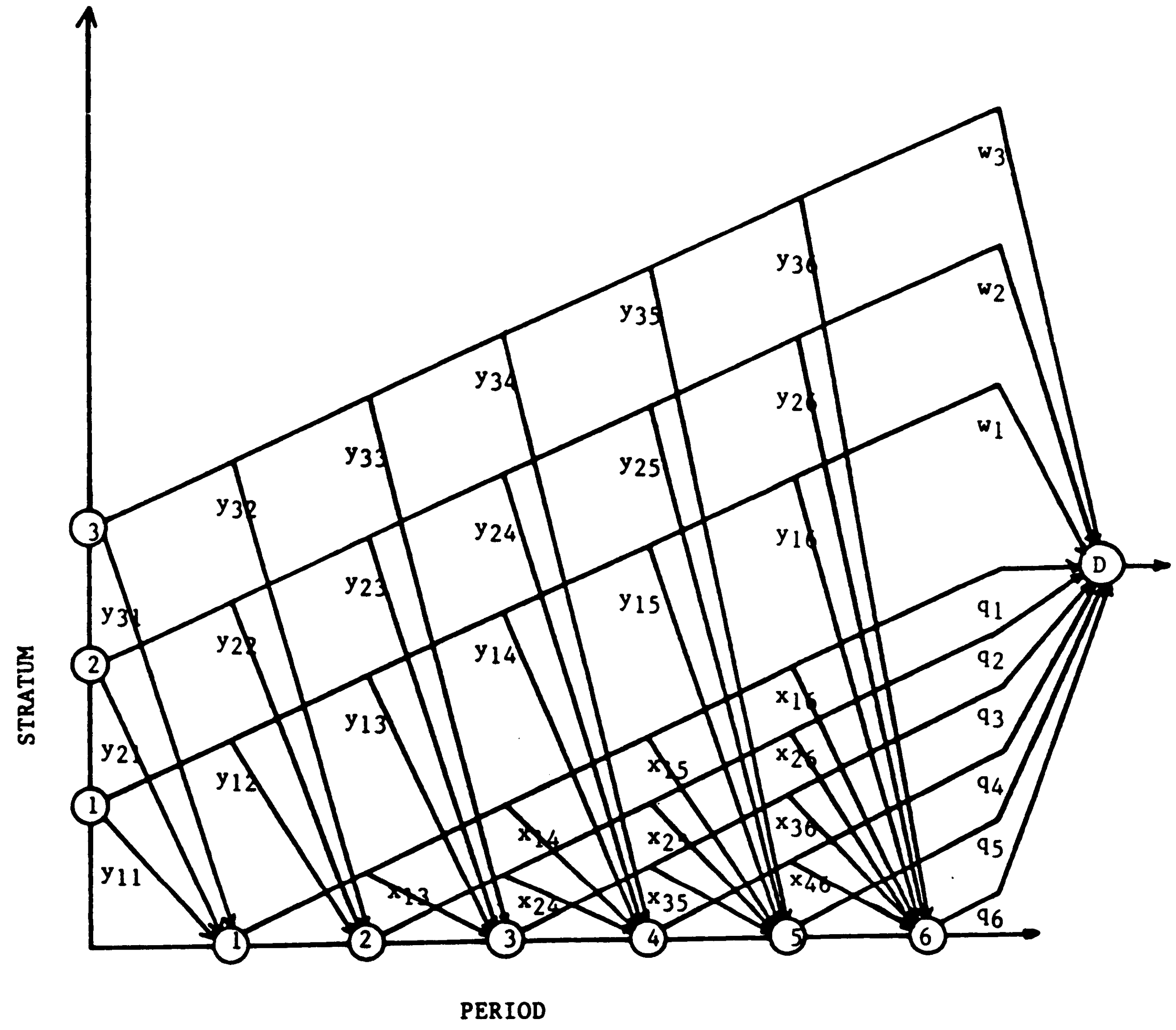

Figure 1. Network graph for Model II-Form 1 with three initial strata, six cutting periods and regeneration lag of 2 periods.

$$\phi_i \geq f_i \quad i = 1, \dots, m$$

$$\gamma_j \geq g_j \quad j = 1, \dots, p$$

$$\phi_i \text{ unrestricted} \quad i = 1, \dots, m$$

$$\gamma_j \text{ unrestricted} \quad j = 1, \dots, p$$

Since minimizing $\sum_{i=1}^{m} b_i \phi_i$ requires the minimum ϕ_i, $i = 1 \dots m$, the solution to the dual problem NFD can be obtained using the following rules:

[1] $\gamma_j = g_j$ for $j = p - L + 1, \dots, p$

[2] $\gamma_j = \max\{\max_{k=j+L\dots p}(d_{jk} + \gamma_k), g_j\}$ for $j = p - L, \dots, 1$ (inductively)

[3] $\phi_i = \max\{\max_{j=1\dots p}(c_{ij} + \gamma_j), f_i\}$ for $i = 1 \dots m$

Also, the complementary slackness conditions are given as follows.

[4] $y_{ij}(\phi_i - c_{ij} - \gamma_j) = 0$ for $i = 1, \dots, m$ and $j = 1, \dots, p$

[5] $x_{jk}(\gamma_j - d_{jk} - \gamma_k) = 0$ for $j = 1, \dots, p - L$ and $k = j + L, \dots, p$

[6] $w_i(\phi_i - f_i) = 0$ for $i = 1, \dots, m$

[7] $q_j(\gamma_j - g_j) = 0$ for $j = 1, \dots, p$

Longest Path Algorithm (LPA)

Following the primal problem, the dual problem and the complementary slackness conditions, a generic statement of the algorithm for network constraints is presented below.

<u>Step 1</u> (Determination of Dual Solution).

(i) For $j = p - L + 1, \ldots, p$, let $\gamma_j = g_j$

(ii) For $j = p - L, \ldots, 1$ (inductively), let $\gamma_j = \max\{\max_{k=j+L \ldots p} (d_{jk} + \gamma_k), g_j\}$

(iii) For $i = 1, \ldots, m$, let $\phi_i = \max\{\max_{j=1 \ldots p}(c_{ij} + \gamma_j), f_i\}$

Step 2 (Determination of Primal Solution).

(i) For $i = 1, \ldots, m$, if $\phi_i = f_i$,

then $w_i = b_i$ and $y_{ij} = 0$, for $j = 1, \ldots, p$.

Otherwise, if $\phi_i = c_{i\bar{j}} + \gamma_{\bar{j}} = \max_{j=1,\ldots,p}\{c_{ij} + \gamma_j\}$,

then $y_{i\bar{j}} = b_i$ and $w_i = 0, y_{ij} = 0$, for $j = 1, \ldots, p$ and $j \neq \bar{j}$.

(ii) For $j = 1, \ldots, p$, if $\gamma_j = g_j$,

then $q_j = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij}$ and $x_{jk} = 0$, for $k = j + L, \ldots, p$.

Otherwise, if $\gamma_j = d_{j\bar{k}} + \gamma_{\bar{k}} = \max_{k=j+L,\ldots,p}\{d_{jk} + \gamma_k\}$,

then $x_{j\bar{k}} = \sum_{i=1}^{m} y_{ij} + \sum_{i=1}^{j-L} x_{ij}$ and $q_j = 0, x_{jk} = 0$, for $k = j + L, \ldots, p$ and $k \neq \bar{k}$.

THEOREM 1. The recursive Algorithm LPA constructs an optimal pair of primal and dual solutions for Problem NFP.

PROOF. From the network graph in Figure 1, ϕ_i, $i = 1, \ldots, m$, are the dual variables for the nodes associated with the initial strata and $\gamma_j, j = 1, \ldots, p$, are the dual variables for the nodes associated with the cutting periods.

Assume that a solution $(y^*, x^*, w^*, q^*, \gamma^*, \phi^*)$ is directly obtained from Step 1 and Step 2. For some initial stratum i, if $\phi_i^* = f_i$, then the stratum is left uncut through the planning horizon and so $w_i^* = b_i$ and $y_{ij}^* = 0, j = 1, \ldots, p$. Otherwise, if $\phi_i^* = c_{i\bar{j}} + \gamma_{\bar{j}} = \max_{j=1,\ldots,p}\{c_{ij} + \gamma_j\}$, then the corresponding decision variable $y_{i\bar{j}}^* = b_i$ and the remaining decision variables, $y_{ij}^*, j \neq \bar{j}$, and w_i^* for this node i are taken as zeros. Hence primal feasibility with respect to the first primal constraint and $y \geq 0, w \geq 0$, and complementary slackness with respective to the first and second dual constraints are satisfied.

For the cutting periods $j, j = 1, \ldots, p$, if $\gamma_j^* = g_j$, then $q_j^* = \sum_{i=1}^{m} y_{ij}^* + \sum_{i=1}^{j-L} x_{ij}^*$ and $x_{jk}^* = 0, k = j + L, \ldots, p$. Otherwise, if $\gamma_j^* = d_{j\bar{k}} + \gamma_{\bar{k}} = \max_{k=j+L,\ldots,p}\{d_{jk} + \gamma_k\}$, then $x_{j\bar{k}}^* = \sum_{i=1}^{m} y_{ij}^* + \sum_{i=1}^{j-L} x_{ij}^*$ and the remaining decision variables, x^*_{jk}, $k \neq \bar{k}$ are taken as zeros. Again primal feasibility with respect to the second primal constraint and $x \geq 0$, $q \geq 0$, and complementary slackness with respect to the third and fourth dual constraints are satisfied.

Therefore, (y^*, x^*, w^*, q^*) solves the primal problem and (γ^*, ϕ^*) solves the dual problem since these solutions satisfy primal and dual feasibilities and the complementary slackness conditions. (Q.E.D.)

This algorithm essentially finds the longest path from each node $i = 1, \ldots, m$ to the dummy node. The quantities $\phi_1, \ldots, \phi_m$ are the values of these longest paths from the nodes $i = 1, \ldots, m$, respectively, and the values $\gamma_1, \ldots, \gamma_p$ are the corresponding longest paths from the nodes $j = 1, \ldots, p$ respectively, to the dummy node. Since the network has a special structure, the above algorithm readily determines these paths, irrespective of the sign on the cost coefficients in the primal problem. Hence, this closed form solution becomes useful even in the context of decomposition/price directive procedures. Also, this algorithm and the solution to the network formulation can be used to verify Berck and Bible's (1984) solution.

In Form 2 of Model I or Model II, the area constraints are specially structure as seen above, while the even-flow constraints can be regarded as complicating coupling constraints. The number of these even-flow constraints equals the number of cutting periods and are typically small. In this case, the Dantzig-Wolfe (1960) decomposition approach can be applied to take advantage of the structure of the area constraints, as suggested by Berck and Bible (1984).

In Form 3 of Model I or Model II, the structure is similar to the even-flow problem, but has an additional number of constraints, including non-declining harvest flow constraints, harvest area constraints, and/or cash flow constraints. In this case, as with Form 2, the constraints other than the area constraints can be treated as complicating side constraints. For example, the Dantzig-Wolfe decomposition method can be applied. Alternatively, since the Form 3 formulation might have a great number of side constraints other than area constraints, the primal-dual conjugate subgradient method following Sen and Sherali (1986) and Sherali and Ulular (1988) may be used. Here, a Lagrangian dual function and a penalty function are first constructed. Then, given some initial pair of primal and dual iterates, the values for the Lagrangian dual function and the penalty function are calculated. If these two values are identical, then optimality is reached. Otherwise, subgradients of the Lagrangian dual function and the penalty function are calculated, a conjugate subgradient based direction is determined, the step length is estimated, and the pair of primal and dual iterates is accordingly updated by taking the corresponding step length and performing a suitable projection.

Chapter V

DESIGN OF SPECIALIZED ALGORITHMS

When only area constraints are included in Model I or Model II of harvest scheduling problems, the formulation exhibits a special structure. In Form 1 of Model I, the area constraints form a separable block diagonal structure, and the problem can be solved via trivial separable knapsack problems. In Form 1 of Model II, the area constraints possess a network structure, which allows one to solve this problem via a network-based method.

When the even-flow constraints are imposed, Form 2 of Model I and Model II can be efficiently solved via the Dantzig-Wolfe decomposition method, taking advantage of the special structure in the area constraints. When more constraints are considered, Form 3 of Model I and Model II becomes complicated and some other solution algorithm might be better to solve these problems. Because of the complexity in terms of the columns in Model I, this study proposes to examine only the Model II formulations. Also, it is of more interest to study the general Model II formulation, rather than only the even-flow constraint variation. So the main part of research is to study the following two competitive algorithms for Model II problems, namely, (i) the Dantzig-Wolfe algorithm and (2) the Primal-Dual Conjugate Subgradient Algorithm. We remark here that Berck and Bible (1984) only present results on an example of Model II-Form 2 solution via the Dantzig-Wolfe algorithm, and do not study in detail the computational behavior of the Dantzig-Wolfe algorithm for this class of problems.

At the final stage of this study, the computational effort will be compared with that for the ordinary simplex method, using several different test problems of Model II-Form 3.

Dantzig-Wolfe Algorithm

The Dantzig-Wolfe algorithm is a price directive approach, in which a direction for modifying the Lagrangian multipliers associated with the complicating coupling constraints is found, and then a suitable step length is taken along this direction. For Model II-Form 3, the primal problem is given as follows.

$$\text{P: Maximize } \sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j$$

$$\text{Subject to } \sum_{j=1}^{p} y_{ij} + w_i = b_i \quad i = 1, \dots, m$$

$$\sum_{k=j+L}^{p} x_{jk} + q_j - \sum_{i=1}^{m} y_{ij} - \sum_{i=1}^{j-L} x_{ij} = 0 \quad j = 1, \dots, p$$

$$\sum_{i=1}^{m} (u_{ij}y_{ij} - u_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} v_{kj}x_{kj} - \sum_{k=1}^{j+1-L} v_{kj+1}x_{kj+1} \le 0 \quad j = 1, \dots, p-1$$

$$\sum_{i=1}^{m} (r_{ij}y_{ij} - r_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} s_{kj}x_{kj} - \sum_{k=1}^{j+1-L} s_{kj+1}x_{kj+1} \le 0 \quad j = 1, \dots, p-1$$

$$y_{ij} \ge 0 \quad i = 1, \dots, m \quad j = 1, \dots, p$$

$$x_{jk} \ge 0 \quad j = 1, \dots, p-L \quad k = j+L, \dots, p$$

$$w_i \ge 0 \quad i = 1, \dots, m$$

$$q_j \geq 0 \quad j = 1, \dots, p$$

As shown above, the first two groups of area constraints have a special network structure, and the remaining volume and cash flow constraints constitute complicating side constraints. The region defined by the area constraints is denoted as X below.

$$X = \{(y,x,w,q): \sum_{j=1}^{p} y_{ij} + w_i = b_i \;\; i = 1, \dots, m, \; \sum_{k=j+L}^{p} x_{jk} + q_j - \sum_{i=1}^{m} y_{ij} - \sum_{i=1}^{j-L} x_{ij} = 0 \;\; j = 1, \dots, p, (y,x,w,q) \geq 0\}$$

It can be shown that X is a non-empty, bounded polyhedron, since the decision variables are bounded through the area constraints and it is feasible to leave all the initial strata uncut through the planning horizon. By the inner linearization manipulation, any element of X can be written as

$$(y,x,w,q) = \sum_{l=1}^{E} \lambda_l (y,x,w,q)^l$$

where $\sum_{l=1}^{E} \lambda_l = 1$

$$\lambda \geq 0$$

and where $(y,x,w,q)^1, (y,x,w,q)^2, \dots, (y,x,w,q)^E$ are the extreme points of X.

The master problem, which is equivalent to the original problem, is defined as

MP: Maximize $\sum_{l=1}^{E} \lambda_l [\sum_{i=1}^{m} \sum_{j=1}^{p} c_{ij} y_{ij}^l + \sum_{j=1}^{p-L} \sum_{k=j+L}^{p} d_{jk} x_{jk}^l + \sum_{i=1}^{m} f_i w_i^l + \sum_{j=1}^{p} g_j q_j^l]$

Subject to $\sum_{l=1}^{E} \lambda_l [\sum_{i=1}^{m} (u_{ij} y_{ij}^l - u_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} v_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} v_{k\,j+1} x_{k\,j+1}^l] \leq 0 \quad j = 1, \dots, p-1$

$$\sum_{l=1}^{E} \lambda_l [\sum_{i=1}^{m} (r_{ij} y_{ij}^l - r_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} s_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} s_{k\,j+1} x_{k\,j+1}^l] \leq 0 \quad j = 1, \dots, p-1$$

$$\sum_{l=1}^{E} \lambda_l = 1$$

$$\lambda_l \geq 0 \qquad l = 1, \ldots, E$$

This master problem, which can be solved by the simplex method, has fewer constraints but many more decision variables than the original problem. The master problem becomes formidable when we introduce all possible extreme points as the decision variables. So a restriction strategy is adopted, which restricts the number of explicitly generated extreme points. That is, restricting $\lambda_l = 0$ for $l \geq k+1$, the restricted master problem (RMP) is as follows.

$$\text{RMP(k): } \bar{z} = \text{Maximize } \sum_{l=1}^{k} \lambda_l [\sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij} y_{ij}^l + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk} x_{jk}^l + \sum_{i=1}^{m} f_i w_i^l + \sum_{j=1}^{p} g_j q_j^l]$$

$$\text{Subject to } \sum_{l=1}^{k} \lambda_l [\sum_{i=1}^{m} (u_{ij} y_{ij}^l - u_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} v_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} v_{k\,j+1} x_{k\,j+1}^l] \leq 0 \quad j = 1, \ldots, p-1$$

$$\sum_{l=1}^{k} \lambda_l [\sum_{i=1}^{m} (r_{ij} y_{ij}^l - r_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} s_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} s_{k\,j+1} x_{k\,j+1}^l] \leq 0 \quad j = 1, \ldots, p-1$$

$$\sum_{l=1}^{k} \lambda_l = 1$$

$$\lambda_l \geq 0 \qquad l = 1, \ldots, k$$

It follows that $\bar{z}$ is a lower bound for the original problem, and if $(\bar{z}, \bar{\lambda})$ solves RMP, then $(\bar{y}, \bar{x}, \bar{w}, \bar{q}) = \sum_{l=1}^{k} (y,x,w,q)^l \bar{\lambda}_l$ is feasible to the original problem. For each nonbasic variable, the corresponding "reduced cost" should be less than or equal to zero if optimality is reached. That is, for $l = 1, \ldots, E$

$$\sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij} y_{ij}^l + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk} x_{jk}^l + \sum_{i=1}^{m} f_i w_i^l + \sum_{j=1}^{p} g_j q_j^l - \sum_{j=1}^{p-1} \bar{\pi}_j [\sum_{i=1}^{m} (u_{ij} y_{ij}^l - u_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} v_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} v_{k\,j+1} x_{k\,j+1}^l]$$

$$- \sum_{j=1}^{p-1} \bar{\phi}_j [\sum_{i=1}^{m} (r_{ij} y_{ij}^l - r_{i\,j+1} y_{i\,j+1}^l) + \sum_{k=1}^{j-L} s_{kj} x_{kj}^l - \sum_{k=1}^{j+1-L} s_{k\,j+1} x_{k\,j+1}^l] - \bar{\pi}_0 \leq 0$$

where $\overline{\pi}$ = the dual variable associated with the first group of constraints

$\overline{\phi}$ = the dual variable associated with the second group of constraints

$\overline{\pi}_0$ = the dual variable associated with the last constraint

The strategy is to find the nonbasic variable with the most positive value above so that we can generate a new extreme point, if necessary. The subproblem, which generates this new extreme point, is given as follows.

$$\begin{aligned} \mathrm{SP}(\overline{\pi}, \overline{\phi}, \overline{\pi}_0):\ \overline{z}_{SP} = -\overline{\pi}_0 + \text{Maximize } & \sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j \\ & - \sum_{j=1}^{p-1}\overline{\pi}_j[\sum_{i=1}^{m}(u_{ij}y_{ij} - u_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} v_{kj}x_{kj} - \sum_{k=1}^{j+1-L} v_{kj+1}x_{kj+1}] \\ & - \sum_{j=1}^{p-1}\overline{\phi}_j[\sum_{i=1}^{m}(r_{ij}y_{ij} - r_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} s_{kj}x_{kj} - \sum_{k=1}^{j+1-L} s_{kj+1}x_{kj+1}] \end{aligned}$$

Subject to $(y,x,w,q) \in X$

If $\overline{z}_{SP} \le 0$, then optimality is reached. Otherwise, suppose that $(\hat{y}, \hat{x}, \hat{w}, \hat{q})$ is one of the extreme points in X which solves this subproblem with $\overline{z}_{SP} > 0$. Then the column corresponding to $(\hat{y}, \hat{x}, \hat{w}, \hat{q})$ is added to RMP, and the procedure returns to solve the RMP for updating the dual variables $(\overline{\pi}, \overline{\phi}, \overline{\pi}_0)$. The main steps for the Dantzig-Wolfe algorithm are then given below.

Initialization . Set $k = 1$ and $(y,x,w,q) = (0,0,b,0)$ as an initial extreme point, since $(0,0,b,0)$ is feasible to X .

Step 1 . Formulate and solve RMP, and generate the corresponding dual solution $(\overline{\pi}, \overline{\phi}, \overline{\pi}_0)$. The objective value $\overline{z}$ of RMP is an incumbent lower bound LB on the problem.

Step 2 . Solve the subproblem $SP(\overline{\pi}, \overline{\phi}, \overline{\pi}_0)$ using algorithm LPA. The value $(\overline{z}_{SP} + \overline{z})$ is an upper bound on the problem. Update the incumbent upper bound UB if this value is smaller.

Step 3 . If $(UB - LB) \leq \varepsilon$, stop; ε – optimality is reached. Otherwise, set $k = k + 1$ and return to Step 1.

Primal-Dual Conjugate Subgradient Algorithm

The primal-dual conjugate subgradient algorithm coordinates a Lagrangian dual function and a primal penalty function to generate a sequence of primal and dual iterates. In order to formulate the Lagrangian dual, define

$$z = (y, x, w, q)$$

$$F \bullet z = \sum_{i=1}^{m}\sum_{j=1}^{p} c_{ij}y_{ij} + \sum_{j=1}^{p-L}\sum_{k=j+L}^{p} d_{jk}x_{jk} + \sum_{i=1}^{m} f_i w_i + \sum_{j=1}^{p} g_j q_j$$

$$A_i \bullet z = \sum_{j=1}^{p} y_{ij} + w_i \quad i = 1, \dots, m$$

$$B_j \bullet z = \sum_{k=j+L}^{p} x_{jk} + q_j - \sum_{i=1}^{m} y_{ij} - \sum_{i=1}^{j-L} x_{ij} \quad j = 1, \dots, p$$

$$G_j \bullet z = \sum_{i=1}^{m}(u_{ij}y_{ij} - u_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} v_{kj}x_{kj} - \sum_{k=1}^{j+1-L} v_{kj+1}x_{kj+1} \quad j = 1, \dots, p-1$$

$$H_j \bullet z = \sum_{i=1}^{m}(r_{ij}y_{ij} - r_{ij+1}y_{ij+1}) + \sum_{k=1}^{j-L} s_{kj}x_{kj} - \sum_{k=1}^{j+1-L} s_{kj+1}x_{kj+1} \quad j = 1, \dots, p-1$$

$$Z = \{z : A_i \bullet z = b_i \; i = 1, \dots, m, \; B_j \bullet z = 0 \; j = 1, \dots, p, \; z \geq 0\}$$

$$\pi = (\delta, \eta)$$

where $F \cdot z$ is the objective function to be maximized; $A_i \cdot z = b_i$ for $i = 1, \ldots, m$ are the first group of constraints; $B_j \cdot z = 0$ for $j = 1, \ldots, p$ are the second group of constraints; $G_j \cdot z \leq 0$ for $j = 1, \ldots, p-1$ are the third group of constraints; $H_j \cdot z \leq 0$ for $j = 1, \ldots, p-1$ are the fourth group of constraints; Z is the set of network-structured constraints; $\delta_j \geq 0$ for $j = 1, \ldots, p-1$ are the Lagrange multipliers associated with the third group of constraints; and $\eta_j \geq 0$ for $j = 1, \ldots, p-1$ are the Lagrange multipliers associated with the fourth group of constraints. Also, let $A, B, G,$ and H denote matrices with respective rows A_i for $i = 1, \ldots, m$, B_j for $j = 1, \ldots, p$, G_j for $j = 1, \ldots, p-1$, and H_j for $j = 1, \ldots, p-1$. The Lagrangian dual problem can be defined as follows.

<u>LD</u>: $\text{Minimize}\{L(\pi) : \pi = (\delta, \eta) \geq 0\}$

where the Lagrangian dual function is given by

$$L(\pi) = \text{Maximum}\{F \cdot z - \delta G \cdot z - \eta H \cdot z : z \in Z\}$$

Note that $L(\pi)$ can be evaluated via the closed-form algorithm LPA. Also, a formula for a subgradient ξ of $L(\cdot)$ at any $\pi = \pi^k \geq 0$ is given below. If $z^k \in Z$ solves for $L(\pi^k)$, then

$$\xi^k = (-G_1 \cdot z^k, \ldots, -G_{p-1} \cdot z^k, -H_1 \cdot z^k, \ldots, -H_{p-1} \cdot z^k)$$

Dual Descent Procedure (DDP)

A conjugate subgradient algorithm is first used to solve the Lagrangian dual function using Sherali and Ulular's (1988) block-halving method and average direction strategy. The major steps for this conjugate subgradient algorithm are given as follows.

Initialization . Put iteration counter $k = 1$, $\pi^1 = (\delta^1, \eta^1) = 0$, and $\text{LB} = F \cdot (0,0,b,0) = \sum_{i=1}^{m} f_i b_i$. Note that LB is a lower bound on the optimal solution and $(0,0,b,0)$ is primal feasible as suggested by Berck and Bible (1984). Compute $L(\pi^1)$ using algorithm LPA and a subgradient ξ of $L(\cdot)$ at $\pi = \pi^1$. Set the incumbent $v^* = L(\pi^1)$, $\pi^* = \pi^1$, $\xi^* = \xi^1$. If $\sum_i \max^2\{0, -\xi_i^*\} < \varepsilon$, stop. Select parameters, $N = 200$, $\bar{v} = 5$, $\bar{\beta} = 2$. Put the consecutive failure counter $t = 0$.

Step 1 . If $k \neq 1, 75,$ or 150, go to Step 2. Otherwise, put $t = 0$, $\pi^k = \pi^*$, $\xi^k = \xi^*$, and compute $\lambda_k = \dfrac{\bar{\beta}[v^* - \text{LB}]}{\|\xi^*\|^2}$. Set $d^k = -\xi^k = -\xi^*$, replace $\bar{\beta}$ by $\bar{\beta}/2$ and go to Step 5.

Step 2 . Compute $L(\pi^k)$ by algorithm LPA and a subgradient ξ^k of $L(\cdot)$ at π^k . If $[\sum_{i:\pi_i^k = 0} \max^2\{0, -\xi_i^k\} + \sum_{i:\pi_i^k > 0} (\xi_i^k)^2] < \varepsilon$, then stop with π^k as (near) optimal. If $L(\pi^k) < v^*$, put $v^* = L(\pi^k)$, $\pi^* = \pi^k$, $\xi^* = \xi^k$, $t = 0$ and go to Step 4. Else, proceed to Step 3.

Step 3 . Replace t by $t + 1$. If $t < \bar{v}$, go to Step 4. Otherwise, put $t = 0$, $\lambda_k = \dfrac{\lambda_{k-1}}{2}$, $\pi^k = \pi^*$, $\xi^k = \xi^*$, $d^k = -\xi^*$ and go to Step 5.

Step 4 . Put $\lambda_k = \lambda_{k-1}$, $d^k = -\xi^k + \dfrac{\|\xi^k\|}{\|d_a^{k-1}\|} d_a^{k-1}$. Proceed to Step 5.

Step 5 . Compute $\pi_i^{k+1} = \max\{0, \pi_i^k + \lambda_k d_i^k\}$ for all i and $d_a^k = (\pi^{k+1} - \pi^k)$ = actual direction of motion. Increment k by one. If $k > N$, stop. Else, return to Step 1.

Primal Penalty Function Maximization

Given $\pi^* = (\delta^*, \eta^*)$ as found via the above procedure DDP, a primal penalty function is now maximized in order to determine a primal solution. Consider the problem $L(\pi^*) = \text{Maximum}\{F \cdot z - \delta^* G \cdot z - \eta^* H \cdot z \; : \; z \in Z\}$. Using algorithm LPA, let $z^* \in Z$ solve this problem, and let ϕ^*, γ^* be the optimal dual multipliers associated with the area constraints in Z.

Note that if z^* is feasible to constraint sets 3 and 4, then z^* is optimal; stop. Else, consider the following problem:

P: Maximize $F \bullet z$

Subject to $A \bullet z = b$

$B \bullet z = 0$

$G \bullet z \leq 0$

$H \bullet z \leq 0$

$z \geq 0$

The dual problem to P is

D: Minimize $b'\phi$

Subject to $A'\phi + B'\gamma + G'\delta + H'\eta \geq F$

ϕ, γ unrestricted

$\delta, \eta \geq 0$

In expanded form, this problem is given below.

D: Minimize $\sum_{i=1}^{m} b_i \phi_i$

Subject to $\phi_i - \gamma_j - u_{ij}\delta_{j-1} + u_{ij}\delta_j - r_{ij}\eta_{j-1} + r_{ij}\eta_j \geq c_{ij}$ for $i = 1, \dots, m, j = 1, \dots, p$

$\gamma_j - \gamma_k - v_{jk}\delta_{k-1} + v_{jk}\delta_k - s_{jk}\eta_{k-1} + s_{jk}\eta_k \geq d_{jk}$ for $j = 1, \dots ,p - L, k = j + L, \dots, p$

$\phi_i \geq f_i$ for $i = 1, \dots, m$

$\gamma_j \geq g_j$ for $j = 1, \dots, p$

ϕ_i unrestricted for $i = 1, \dots, m$

γ_j unrestricted for $j = 1, \dots, p$

$\delta_j \geq 0$ for $j = 1, \dots, p - 1$

$\eta_j \geq 0$ for $j = 1, \dots, p - 1$

Note that the vector $(\phi^*, \gamma^*, \delta^*, \eta^*)$ is the prescribed dual solution to this problem. The complementary slackness conditions with respect to this solution are given below.

$$\delta_j^*[\sum_{i=1}^{m}(u_{ij}y_{ij} - u_{i\,j+1}y_{i\,j+1}) + \sum_{k=1}^{j-L} v_{kj}x_{kj} - \sum_{k=1}^{j+1-L} v_{k\,j+1}x_{k\,j+1}] = 0 \quad \text{for } j = 1, \ldots, p-1$$

$$\eta_j^*[\sum_{i=1}^{m}(r_{ij}y_{ij} - r_{i\,j+1}y_{i\,j+1}) + \sum_{k=1}^{j-L} s_{kj}x_{kj} - \sum_{k=1}^{j+1-L} s_{k\,j+1}x_{k\,j+1}] = 0 \quad \text{for } j = 1, \ldots, p-1$$

$$y_{ij}[\phi_i^* - \gamma_j^* - u_{i\,j}\delta_{j-1}^* + u_{ij}\delta_j^* - r_{i\,j}\eta_{j-1}^* + r_{ij}\eta_j^* - c_{ij}] = 0 \quad \text{for } i = 1, \ldots, m, \;\; j = 1, \ldots, p$$

$$x_{jk}[\gamma_j^* - \gamma_k^* - v_{j\,k}\delta_{k-1}^* + v_{jk}\delta_k^* - s_{j\,k}\eta_{k-1}^* + s_{jk}\eta_k^* - d_{jk}] = 0 \quad \text{for } j = 1, \ldots, p-L, \;\; k = j+L, \ldots, p$$

$$w_i[\phi_i^* - f_i] = 0 \quad \text{for } i = 1, \ldots, m$$

$$q_j[\gamma_j^* - g_j] = 0 \quad \text{for } j = 1, \ldots, p$$

Hence, we only need to find a feasible solution to P which satisfies the above complementary slackness conditions in order to get an optimal primal solution, given that $(\phi^*, \gamma^*, \delta^*, \eta^*)$ is dual optimal. However, since $(\phi^*, \gamma^*, \delta^*, \eta^*)$ may only be a near optimal dual feasible solution, we use a tolerance $\varepsilon > 0$ to partially enforce the complementary slackness relationships as noted below.

[8] For $i = 1, \ldots, m, j = 1, \ldots, p,$ if $\phi_i^* - \gamma_j^* - u_{i\,j}\delta_{j-1}^* + u_{ij}\delta_j^* - r_{i\,j}\eta_{j-1}^* + r_{ij}\eta_j^* - c_{ij} \geq \varepsilon$,
then set $y_{ij} = 0$ and let this $(i,j) \in J_1$.

[9] For $j = 1, \ldots, p-L, k = j+L, \ldots, p,$ if $\gamma_j^* - \gamma_k^* - v_{j\,k}\delta_{k-1}^* + v_{jk}\delta_k^* - s_{j\,k}\eta_{k-1}^* + s_{jk}\eta_k^* - d_{jk} \geq \varepsilon$,
then set $x_{jk} = 0$ and let this $(j,k) \in J_2$.

[10] For $i = 1, \ldots, m,$ if $\phi_i^* - f_i \geq \varepsilon$, then set $w_i = 0$ and let this $i \in J_3$.

[11] For $j = 1, \dots, p$, if $\gamma_j^* - g_j \geq \varepsilon$, then set $q_j = 0$ and let this $j \in J_4$.

[12] For $j = 1, \dots, p-1$, if $\delta_j^* \geq \varepsilon$,

then set $\sum_{i=1}^{m}(u_{ij}y_{ij} - u_{i\,j+1}y_{i\,j+1}) + \sum_{k=1}^{j-L} v_{kj}x_{kj} - \sum_{k=1}^{j+1-L} v_{k\,j+1}x_{k\,j+1} = 0$ and let this $j \in I_1$

[13] For $j = 1, \dots, p-1$, if $\eta_j^* \geq \varepsilon$,

then set $\sum_{i=1}^{m}(r_{ij}y_{ij} - r_{i\,j+1}y_{i\,j+1}) + \sum_{k=1}^{j-L} s_{kj}x_{kj} - \sum_{k=1}^{j+1-L} s_{k\,j+1}x_{k\,j+1} = 0$ and let this $j \in I_2$

Now, let $\bar{J}_1, \bar{J}_2, \bar{J}_3, \bar{J}_4, \bar{I}_1$ and $\bar{I}_2$ be the complements of J_1, J_2, J_3, J_4, I_1 and I_2, respectively. Then we can set up the primal problem in the reduced space as follows.

Maximize $\displaystyle \sum_{(i,j)\in \bar{J}_1} c_{ij}y_{ij} + \sum_{(j,k)\in \bar{J}_2} d_{jk}x_{jk} + \sum_{i\in \bar{J}_3} f_i w_i + \sum_{j\in \bar{J}_4} g_j q_j$

Subject to $\displaystyle \sum_{j:(i,j)\in \bar{J}_1} y_{ij} + w_i = b_i \quad i \in \bar{J}_3$

$$\sum_{j:(i,j)\in \bar{J}_1} y_{ij} = b_i \quad i \in J_3$$

$$\sum_{k\geq j+L:(j,k)\in \bar{J}_2} x_{jk} + q_j - \sum_{i:(i,j)\in \bar{J}_1} y_{ij} - \sum_{i\leq j-L:(i,j)\in \bar{J}_2} x_{ij} = 0 \quad j \in \bar{J}_4$$

$$\sum_{k\geq j+L:(j,k)\in \bar{J}_2} x_{jk} - \sum_{i:(i,j)\in \bar{J}_1} y_{ij} - \sum_{i\leq j-L:(i,j)\in \bar{J}_2} x_{ij} = 0 \quad j \in J_4$$

$$\sum_{i:(i,j)\in \bar{J}_1} u_{ij}y_{ij} - \sum_{i:(i,j+1)\in \bar{J}_1} u_{i\,j+1}y_{i\,j+1} + \sum_{k\leq j-L:(k,j)\in \bar{J}_2} v_{kj}x_{kj} - \sum_{k\leq j+1-L:(k,j+1)\in \bar{J}_2} v_{k\,j+1}x_{k\,j+1} = 0 \quad j \in I_1$$

$$\sum_{i:(i,j)\in \bar{J}_1} u_{ij}y_{ij} - \sum_{i:(i,j+1)\in \bar{J}_1} u_{i\,j+1}y_{i\,j+1} + \sum_{k\leq j-L:(k,j)\in \bar{J}_2} v_{kj}x_{kj} - \sum_{k\leq j+1-L:(k,j+1)\in \bar{J}_2} v_{k\,j+1}x_{k\,j+1} \leq 0 \quad j \in \bar{I}_1$$

$$\sum_{i:(i,j)\in \bar{J}_1} r_{ij}y_{ij} - \sum_{i:(i,j+1)\in \bar{J}_1} r_{i\,j+1}y_{i\,j+1} + \sum_{k\leq j-L:(k,j)\in \bar{J}_2} s_{kj}x_{kj} - \sum_{k\leq j+1-L:(k,j+1)\in \bar{J}_2} s_{k\,j+1}x_{k\,j+1} = 0 \quad j \in I_2$$

$$\sum_{i:(i,j)\in \bar{J}_1} r_{ij}y_{ij} - \sum_{i:(i,j+1)\in \bar{J}_1} r_{i\,j+1}y_{i\,j+1} + \sum_{k\leq j-L:(k,j)\in \bar{J}_2} s_{kj}x_{kj} - \sum_{k\leq j+1-L:(k,j+1)\in \bar{J}_2} s_{k\,j+1}x_{k\,j+1} \leq 0 \quad j \in \bar{I}_2$$

$$(y,x,w,q) \geq 0$$

Denoting the vector of surviving variables as

$$z' = (y_{ij} \text{ for } (i,j) \in \bar{J}_1,\ x_{jk} \text{ for } (j,k) \in \bar{J}_2,\ w_i \text{ for } i \in \bar{J}_3,\ q_j \text{ for } j \in \bar{J}_4),$$

we can abbreviate the above problem using obvious notation as follows.

$$\begin{aligned} &\text{Maximize } F \cdot z' \\ &\text{Subject to } g_r(z') \leq 0 \text{ for } r = 1, \ldots, R && [14] \\ &\qquad h_k(z') = 0 \text{ for } k = 1, \ldots, K && [15] \\ &\qquad z' \geq 0 \end{aligned}$$

where $g_r(z') \leq 0$ for $r = 1, \ldots, R$ and $h_k(z') = 0$ for $k = 1, \ldots, K$ respectively represent the reduced set of inequalities and the equalities in the problem. Note that $R = |\bar{I}_1| + |\bar{I}_2|$ and $K = m + p + |I_1| + |I_2|$. Correspondingly, let us rename the dual solution $(\phi^*, \gamma^*, \delta^*, \eta^*)$ as α_r^* for $r = 1, \ldots, R$, and β_k^* for $k = 1, \ldots, K$, associated with the constraints [14] and [15], respectively.

We now construct a primal penalty function for this problem as in Sherali and Ulular (1988). This function will then be maximized using a conjugate subgradient algorithm similar to the procedure DDP above. Note that because of the use of the complementary slackness conditions, optimal and feasible solutions act in concert here.

The proposed penalty function is given below.

$$f(z') = F \cdot z' - \frac{1}{2}\sum_{r=1}^{R} \mu_r \max{}^2\{g_r(z') + \frac{\alpha_r^*}{\mu_r}, 0\} + \frac{1}{2}\sum_{r=1}^{R} \frac{\alpha_r^{*2}}{\mu_r}$$

$$- \sum_{k=1}^{K} \beta_k^* h_k(z') - \frac{1}{2}\sum_{k=1}^{K} \rho_k h_k^2(z')$$

$$- \sum_{r=1}^{R} \mu_r \max\{g_r(z'),0\} - \sum_{k=1}^{K} \rho_k |h_k(z')| \ ,$$

where the penalty parameters μ_r for $r = 1, \dots ,R$ and ρ_k for $k = 1, \dots ,K$ are given by

$$\mu_r = \alpha_r^* + \mu_r^0 \quad \text{for } r = 1, \dots , R$$

and

$$\rho_k = \beta_k^* + \rho_k^0 \quad \text{for } k = 1, \dots ,K$$

and where each of μ_r^0 for $r = 1, \dots , R$ and ρ_k^0 for $k = 1, \dots , K$ is initialized at one, and is doubled every 10-20 iterations if the incumbent solution violates the corresponding constraint by more than a specified tolerance $\tau > 0$.

We now state the conjugate subgradient algorithm for maximizing $f(z')$ over $z' \geq 0$. For this purpose, note that a subgradient of $f(\bullet)$ at z' is given as follows.

$$\zeta^* = F' - \sum_{r=1}^{R} \mu_r \max\{g_r(z') + \frac{\alpha_r^*}{\mu_r}, 0\} \nabla g_r(z') - \sum_{k=1}^{K} (\beta_k^* + \rho_k h_k(z')) \nabla h_k(z')$$

$$- \sum_{r:g_r(z')>0} \mu_r \nabla g_r(z') - \sum_{k:h_k(z')>0} \rho_k \nabla h_k(z') + \sum_{k:h_k(z')<0} \rho_k \nabla h_k(z')$$

where F' is the coefficient vector associated with the objective function in the reduced space; $g_r(z')$ is the r^{th} constraint in equation [14]; $\nabla g_r(z')$ is the subgradient associated with $g_r(z')$; $h_k(z')$ is the k^{th} constraint in equation [15]; and $\nabla h_k(z')$ is the subgradient associated with $h_k(z')$.

The major steps for the conjugate subgradient algorithm are given below.

Initialization . Put iteration counter $k = 1, z'^k = z^{*\prime}$, and $UB = L(\pi^*)$. Compute $f(z'^k)$ and a subgradient ζ^k of $f(\bullet)$ at $z' = z'^k$. Set the incumbent $v^* = f(z'^k), z'^* = z'^k, \zeta^* = \zeta^k$. If

$\sum_j \max^2\{0, \zeta_j^*\} < \varepsilon$, stop. Select parameters, $N = 200, \bar{v} = 5, \bar{\beta} = 2$. Put the consecutive failure counter $t = 0$.

Step 1 . If $k \neq 1, 75$, or 150, go to Step 2. Otherwise, put $t = 0, z'^k = z'^*, \zeta^k = \zeta^*$, and compute $\lambda_k = \dfrac{\bar{\beta}[UB - v^*]}{\|\zeta^*\|^2}$. Set $D^k = \zeta^k = \zeta^*$, replace $\bar{\beta}$ by $\bar{\beta}/2$ and go to Step 5.

Step 2 . Compute $f(z'^k)$ and a subgradient ζ^k of $f(\bullet)$ at z'^k . If $[\sum_{j:z_j'^k=0} \max^2\{0, \zeta_j^k\} + \sum_{j:z_j'^k>0} (\zeta_j^k)^2] < \varepsilon$, then stop with z'^k as (near) optimal. If $f(z'^k) > v^*$, put $v^* = f(z'^k), z'^* = z'^k, \zeta^* = \zeta^k, t = 0$ and go to Step 4. Else, proceed to Step 3.

Step 3 . Replace t by $t + 1$. If $t < \bar{v}$, go to Step 4. Otherwise, put $t = 0, \lambda_k = \dfrac{\lambda_{k-1}}{2}$, $z'^k = z'^*, \zeta^k = \zeta^*, D^k = \zeta^*$ and go to Step 5.

Step 4 . Put $\lambda_k = \lambda_{k-1}, D^k = \zeta^k + \dfrac{\|\zeta^k\|}{\|D_a^{k-1}\|} D_a^{k-1}$. Proceed to Step 5.

Step 5 . Compute $z_j'^{k+1} = \max\{0, z_j'^k + \lambda_k D_j^k\}$ for all j and $D_a^k = (z'^{k+1} - z'^k)$ = actual direction of motion. Increment k by one. If $k > N$, stop. Else, return to Step 1.

Chapter VI

COMPUTATIONAL RESULTS

In this chapter, computational experience is presented with Algorithm LPA and the revised simplex method for the Model II-Form 1 problems, and with the Dantzig-Wolfe (D-W) algorithm, the Primal-Dual Conjugate Subgradient Algorithm (PDCSA) and the revised simplex algorithm for the general Model II-Form 3 problems. The computer source codes for Algorithm LPA, the D-W algorithm, and PDCSA are written in FORTRAN 77 and run on an IBM 3090 mainframe system. The Mathematical Programming System (MPS III) software library is used to run the revised simplex algorithm. The overall computational performance for Algorithm LPA and for the D-W algorithm and PDCSA are compared with that for the standard linear programming package, MPS III, for the respective problems.

TEST PROBLEMS

The test problems are generated using the yield equation (Clutter et al. 1983)

$$Y_t = 100(1 - e^{-0.05t})^2 \quad \text{for} \quad t \geq 10$$

where t is the stand age in years and Y_t is the merchantable volume at age t in cunits per acre. One cunit is about 3 cubic feet. The stumpage price is assumed to be $25 per cunit throughout the planning horizon. The interest rate is assumed to be 5 percent annually. This yield equation is used to estimate the cunits per acre in each cutting period when cutting units are harvested. The value of cunits per acre is then assigned to the coefficients of the non-declining harvest volume constraints. The revenue per acre for each cutting period is the product of the cunits per acre times the stumpage price. The value of revenue per acre is assigned to the coefficients of the cash flow constraints. The interest rate is used to generate the net present value for the coefficients of the objective function. The problem size depends on the length of the planning period and the number of initial strata considered. The descriptors for the test problems are shown in Table 3. The number of network constraints (NC) equals the total number of area constraints. The number of side constraints (SC) equals the total number of non-area constraints. For the a-type problems, the side constraints involve only the non-declining harvest volume constraints and for the b-type problems, both the non-declining harvest volume and the cash flow constraints are included. So the b-type problems have twice the number of side constraints as the a-type problems. For problems 1 and 2, we fix the number of initial strata and vary the number of planning periods. For problems 3, 4 and 5 or problems 6 and 7, we fix the number of planning periods and vary the number of initial strata. The aim is to examine the effect of these variations on the overall computational effort.

COMPUTATIONAL EXPERIENCE WITH ALGORITHM LPA

Algorithm LPA can be used to solve harvest scheduling problems involving only the network area constraints. This is the first step in finding a solution to the Model II-Form 3 problems. This solution provides a maximum likelihood solution for the harvest scheduling problem, and can be

Table 3. Harvest scheduling problems tested.

Problem No.	p	m	L	Var	NC	SC
1a	8	2	5	32	10	7
1b	8	2	5	32	10	14
2a	16	2	5	116	18	15
2b	16	2	5	116	18	30
3a	10	10	3	148	20	9
3b	10	10	3	148	20	18
4a	10	15	3	203	25	9
4b	10	15	3	203	25	18
5a	10	20	3	258	30	9
5b	10	20	3	258	30	18
6a	20	10	5	350	30	19
6b	20	10	5	350	30	38
7a	20	15	5	455	35	19
7b	20	15	5	455	35	38

p = number of planning periods.

m = number of initial strara.

L = regeneration lag.

Var = number of decision variables.

NC = number of network constraints.

SC = number of side constraints.

useful for decision making. For instance, it indicates the loss in objective function value due to imposed side constraints. An illustrative example for Algorithm LPA is presented below for a simple problem.

<u>Example.</u> A harvest scheduling problem with $p = 5, m = 2$ and $L = 3$ is shown below.

$$\begin{aligned}
\text{Max}\quad & 997y_{11} + 926y_{12} + 820y_{13} + 551y_{14} + 590y_{15} + 1180y_{21} + \\
& 1047y_{22} + 898y_{23} + 754y_{24} + 622y_{25} + 262x_{14} + 295x_{15} + \\
& 205x_{25} + 710w_1 + 742w_2 + 415q_1 + 325q_2 + 234q_3 + 156q_4 + 125q_5 \\
\text{S. T.}\quad & y_{11} + y_{12} + y_{13} + y_{14} + y_{15} + w_1 = 90 \\
& y_{21} + y_{22} + y_{23} + y_{24} + y_{25} + w_2 = 65 \\
& -y_{11} - y_{21} + x_{14} + x_{15} + q_1 = 0 \\
& -y_{12} - y_{22} + x_{25} + q_2 = 0 \\
& -y_{13} - y_{23} + q_3 = 0 \\
& -y_{14} - y_{24} - x_{24} + q_4 = 0 \\
& -y_{15} - y_{25} - x_{15} - x_{25} + q_5 = 0 \\
& y \geq 0, x \geq 0, w \geq 0, q \geq 0
\end{aligned}$$

The following steps give the optimal dual solution associated with the network area constraints.

(i) For $j = p - L + 1, \dots, p,$

$$\gamma_3 = 234$$

$$\gamma_4 = 156$$

$$\gamma_5 = 125$$

(ii) For $j = p - L, \dots, 1,$

$$\gamma_2 = \max\{205 + 125, 325\} = 330$$

$$\gamma_1 = \max\{262 + 156, 295 + 125, 415\} = 420$$

(iii) For $i = 1, \dots, m,$

$$\phi_1 = \max\{997 + 420, 926 + 330, 820 + 234, 551 + 156, 590 + 125, 710\}$$

$$= 1417$$

$$\phi_2 = \max\{1180 + 420, 1047 + 330, 898 + 234, 754 + 156, 622 + 125, 742\}$$

$$= 1600$$

The optimal primal solution is then given below.

(iv) For $i = 1, \dots, m,$

$\phi_1 = 1417$ implies $y_{11} = 90$

$\phi_2 = 1600$ implies $y_{21} = 65$

(v) For $j = 1, \dots, p,$

$\gamma_1 = 420$ implies $x_{15} = 155$

$\gamma_2 = 330$ implies $x_{25} = 0$

$\gamma_3 = 234$ implies $q_3 = 0$

$\gamma_4 = 156$ implies $q_4 = 0$

$\gamma_5 = 125$ implies $q_5 = 155$

and other variables are zeros.

The source code for Algorithm LPA solves the Model II-Form 1 problems in closed form. The storage requirement is $(pm + \sum_{j=1}^{p-L} \sum_{k=j+L}^{p} + m + p + m)$. Only the coefficients in the objective function and the right-hand-side values need to be stored. It can be expected that the computational time and storage requirements are dramatically reduced when the problem is large. Computational results using LPA and MPS III are shown in Table 4. The number of iterations for MPS III increases steadily with problem size, and so does the computational time. The computational time for LPA is negligible even for the larger sized test problems.

Table 4. Computational results for Algorithm LPA and MPS III respectively.

Problem No.	Number of iterations		CPU time in seconds	
	LPA	MPS III	LPA	MPS III
1	1	15	0.00	0.18
2	1	44	0.00	0.30
3	1	57	0.00	0.36
4	1	72	0.00	0.42
5	1	89	0.00	0.48
6	1	94	0.00	0.54
7	1	108	0.00	0.72

COMPUTATIONAL EXPERIENCE WITH THE D-W ALGORITHM

The implementation of the D-W algorithm takes advantage of Algorithm LPA in the subproblem step and uses the revised simplex method to solve the master problems. The D-W algorithm is assumed to have converged when the gap between the upper bound value and the lower bound value is within a specified percentage tolerance. For each problem solved, we report the total number of major iterations, the number of degenerate pivots, the number of binding side constraints, and the computational Central Processing Unit (CPU) time for the D-W algorithm. A pivot is defined here to be "degenerate" when the lower bound value does not change during the master problem step. The computational results obtained for the D-W algorithm are shown in Table 5 and Table 6. The computational results obtained using MPS III are shown in Table 7.

The computational time and the number of iterations for the D-W algorithm depend on the optimality tolerance selected, that is, the admissible gap between the upper bound value and the lower bound value. When a 1 percent accuracy tolerance is used, the computational time and the number of iterations are much smaller than those using MPS III for the larger sized a-type problems, and are smaller for most of the b-type problems. For problems 3b, 4b and 5b, the number of iterations are respectively 86, 76 and 94 for the D-W algorithm, but 151, 196 and 234 using MPS III. For problems 6b and 7b, the number of iterations are respectively 358 and 462 for the D-W algorithm, but 615 and 595 using MPS III. So, the D-W algorithm takes significantly fewer iterations than does MPS III for the same a-type problems, and for most of the b-type problems. While the computational time for the D-W algorithm is also smaller, the difference is not practically significant for the size of problems solved. However, for significantly larger problems, and for problems having multiple species which result in multiple diagonal blocks, the overall computational advantage of the D-W algorithm over MPS III is likely to become significant. In this study, we did not run larger problems because of the difficulty of manually inputting the data. However, the trend for the results with an increase in problem size is evident from our

Table 5. Computational results for the D-W algorithm using 5 percent tolerance.

Problem No.	ITER	CPU time	NDP	NBSC	ACC
1a	8	0.00	6	7	0.8
1b	22	0.01	19	3	0.1
2a	27	0.06	14	15	2.2
2b	30	0.11	16	15	1.9
3a	24	0.03	9	10	1.9
3b	52	0.13	19	16	0.8
4a	31	0.06	11	10	2.1
4b	45	0.15	20	16	2.1
5a	20	0.05	7	10	2.9
5b	48	0.21	15	16	1.3
6a	56	0.44	18	20	1.3
6b	220	3.07	84	38	1.5
7a	56	0.64	18	19	1.5
7b	176	3.66	61	38	2.1

ITER = number of iterations.

NDP = number of degenerate pivots.

NBSC = number of binding side constraints.

ACC = percent accuracy of the best solution found.

Table 6. Computational results for the D-W algorithm using 1 percent tolerance.

Problem No.	ITER	CPU time	NDP	NBSC	ACC
1a	14	0.00	6	8	0.03
1b	24	0.01	21	3	0.03
2a	49	0.10	14	16	0.14
2b	69	0.27	19	17	0.11
3a	34	0.05	9	10	0.56
3b	86	0.22	19	16	0.37
4a	48	0.09	11	10	0.42
4b	76	0.26	20	16	0.51
5a	44	0.10	7	10	0.37
5b	94	0.40	20	16	0.37
6a	100	0.75	18	20	0.32
6b	358	4.81	84	38	0.30
7a	100	1.09	18	20	0.38
7b	462	9.09	61	38	0.29

ITER = number of iterations.

NDP = number of degenerate pivots.

NBSC = number of binding side constraints.

ACC = percent accuracy of the best solution found.

Table 7. Computational results for MPS III.

Problem No.	Number of iterations	CPU time in seconds
1a	32	0.24
1b	42	0.24
2a	102	0.60
2b	118	0.72
3a	145	0.78
3b	151	0.84
4a	150	0.90
4b	196	1.14
5a	203	1.14
5b	234	1.38
6a	306	1.92
6b	615	5.16
7a	281	1.80
7b	595	5.40

computations, as well as from the expected performance of the simplex algorithm used in MPS III, and as used for solving the master problems in the D-W algorithm.

Berck and Bible's (1984) results showed that the D-W algorithm took only half of the computational time using the simplex algorithm. Part of the reason for this is that they used multiple species which decomposed the problem into multiple separable subproblems for the D-W algorithm.

When a 5 percent accuracy tolerance is used, the computational time and the number of iterations are smaller using the D-W algorithm than using MPS III, for both the a-type and the b-type problems. For problems 3, 4 and 5, the D-W algorithm needs no more than 60 iterations to reach optimality, while the revised simplex method needs more than 145 iterations. Also, the CPU time is smaller for the D-W algorithm.

When a 0.1 percent accuracy tolerance is used for problem 6a, 6b, 7a and 7b, the number of iterations are respectively 186, 914, 283, and 1232 for the D-W algorithm. The CPU times for these problems are respectively 1.45, 12.81, 3.20 and 25.81 seconds. This suggests that the D-W algorithm needs many more iterations and results in a significant increase in computational effort over MPS III if a very small accuracy tolerance is imposed. Note that the actual accuracy of the solutions determined, when a maximum limit of 1 percent or 5 percent is imposed, is significantly below these imposed limits. Table 5 and Table 6 show that the percent accuracy of the best solution found varies from 0.8 to 2.9 percent of the true optimum value for the 5 percent accuracy tolerance, and from 0.03 to 0.56 percent for the 1 percent accuracy tolerance.

Comparison of total iterations for the 1 percent tolerance and the 5 percent tolerance runs suggests that the D-W algorithm makes rapid progress in the initial iterations. For example, the ratios of total iterations using a 5 percent tolerance to those using a 1 percent tolerance for problems 3, 4 and 5 range from 0.5 to 0.7, and for problem 7b, the D-W algorithm takes only 176 iterations to reach within 5 percent of the optimum value, but takes 462 iterations to reach within 1 percent of this value. This implies that although the D-W algorithm makes rapid advances in the initial iterations, it has a very slow asymptotic convergence rate in the final iterations.

When the number of initial strata is fixed, increasing the number of side constraints significantly increases the computational effort for the D-W algorithm but does not significantly increase the computational effort for the revised simplex method. Statistically, the computational time for the general simplex method is approximately proportional to the number of constraints raised to 2.5 to 3.0. When the number of side constraints is fixed, increasing initial strata does not significantly increase the computational effort for the D-W algorithm but significantly increases the computational effort for the revised simplex method. So the D-W algorithm is greatly affected by the number of side constraints, while the revised simplex method is greatly affected by the number of initial strata.

The number of degenerate pivots is almost the same for both the 1 percent and the 5 percent accuracy tolerances, and these occur mainly in the initial stages of the algorithm. For the large sized test problems, the side constraints are almost all binding for both the a-type and b-type problems. For the small sized b-type problems, some of the side constraints are non-binding.

COMPUTATIONAL EXPERIENCE WITH ALGORITHM PDCSA

This section gives the results of running Algorithm PDCSA on the general Model II problems. Algorithm PDCSA is assumed to have converged When the dual objective value or the primal objective value fails to improve within 20 consecutive iterations, the algorithm terminates with the incumbent solution as near optimal. The maximum number of iterations for both the Dual Descent Procedure and the Primal Penalty Function Maximization is 200. For each problem solved, we report the dual objective value, the primal objective value, the CPU time in seconds, and the optimum value, obtained via MPS III.

Table 8 gives the computational results for the test problems. Computational results show that if the number of side constraints is doubled, the CPU time increases three to six times. If the number of side constraints is fixed, increasing the number of decision variables does not significantly increase the computational effort. This implies that Algorithm PDCSA is greatly affected by the number of side constraints but not significantly influenced by the number of decision variables. The computational time is smaller for the a-type problems but larger for the b-type problems, compared with those using MPS III. Also, the gap between the dual objective value and the primal objective value is large for most of the b-type problems. This duality gap might reduce if scaling strategies and suitable step-sized and projection strategies are used, as in Sherali and Ulular (1988).

COMPARISON OF STORAGE REQUIREMENTS

The storage requirement for the D-W algorithm is far less than that for the revised simplex method. Only the coefficients in the objective function, the side constraints, and the right-hand-side values need to be stored. Additionally, when implementing the revised simplex method to solve the master problems, we also need storage for the working basis and the entering column. The total storage requirement for the D-W algorithm is given below.

$$STORAGE(D-W) = (pm + \sum_{j=1}^{p-L} \sum_{k=j+L}^{p})(p_n + 2) + 4(m + p) + (p_n + 2) + (p_n + 2)^2$$

where p_n is the number of the side constraints. For the revised simplex method, all the coefficients in the objective function, the network constraints, the side constraints, and the right-hand-side values need to be stored. Also, we need additional storage for the working basis and the entering column. The storage requirement for the revised simplex method is given below.

Table 8. Computational results for PDCSA.

Problem No.	Optimum obj.	$\frac{\text{Dual obj.}}{\text{Opt. obj.}}$	No. of iters	$\frac{\text{Primal obj.}}{\text{Opt. obj.}}$	Average infeas.	No. of iters	CPU time in seconds
1a	264807	1.002	85	0.999	0/17	94	0.1
1b	228725	1.171	85	0.958	0/24	94	0.6
2a	198370	1.046	84	0.970	0/33	94	0.6
2b	198370	1.046	84	0.970	0/48	94	1.8
3a	495495	1.034	85	0.981	0/29	94	0.4
3b	445614	1.230	89	0.926	9/38	200	1.6
4a	893258	1.009	90	0.993	0/34	94	0.6
4b	678404	1.021	94	0.993	9/43	156	2.0
5a	135788	1.007	93	0.994	0/39	94	0.7
5b	1115330	1.124	88	0.968	9/48	200	2.6
6a	545051	1.144	94	0.973	0/49	94	1.8
6b	536782	1.181	94	0.953	19/68	200	8.0
7a	874016	1.103	90	0.942	0/54	94	2.9
7b	781291	1.184	93	0.911	21/73	200	12.0

Average infeas. = Total infeas./Total No. of constraints.

$$STORAGE(RS) = (pm + \sum_{j=1}^{p-L} \sum_{k=j+L}^{p})(p + m + p_n + 1) + (m + p)(p_n + 3) + (p_n + 2)^2$$

For the Primal-Dual Conjugate Subgradient Algorithm, the coefficients in the objective function, the side constraints and the right-hand-side values need to be stored. Also, we need more storage for the subgradient vector, the direction vector, and the actual previous direction vector. The storage requirement for the Primal-Dual Conjugate Subgradient Algorithm is given below.

$$STORAGE(PDCSA) = (pm + \sum_{j=1}^{p-L} \sum_{k=j+L}^{p})(p_n + 4) + 4(m + p)$$

The relative storage requirements for these three competitive methods are shown in Table 9 for the test problems of Table 3. For most of the test problems, the D-W algorithm and the Primal-Dual Conjugate Subgradient Algorithm need no more than half of the storage required for the revised simplex method. If the number of side constraints is relatively small, the relative storage requirements for the D-W algorithm and the Primal-Dual Conjugate Subgradient Algorithm become smaller.

Table 9. The relative storage requirements of the test problems for the revised simplex method, the D-W algorithm and PDCSA.

Problem No.	Simplex	D-W	PDCSA
1a	1.0	0.57	0.49
1b	1.0	0.68	0.45
2a	1.0	0.52	0.49
2b	1.0	0.66	0.53
3a	1.0	0.39	0.41
3b	1.0	0.52	0.49
4a	1.0	0.33	0.36
4b	1.0	0.52	0.51
5a	1.0	0.29	0.32
5b	1.0	0.42	0.42
6a	1.0	0.43	0.43
6b	1.0	0.58	0.54
7a	1.0	0.39	0.40
7b	1.0	0.58	0.56

Chapter VII

CONCLUSIONS

This study has examined problem formulation, structure exploitation and the design of specialized algorithms for solving general harvest scheduling problems. We show that Model I-Form 1 and Model II-Form 1 possess a network structure. In Model I-Form 1, the network constraints form a separable block diagonal structure, which permits one to solve for the decision variables belonging to each individual area constraints independently as a knapsack problem. In Model II-Form 1, the Longest Path Algorithm (LPA) is developed and coded to solve the problem. This algorithm can solve these problems in closed form, with a negligible amount of effort and with a minimal storage requirement. Hence, it is clearly superior to the revised simplex method.

The Dantzig-Wolfe (D-W) algorithm is coded and tuned to solve general Model II problems, taking advantage of Algorithm LPA and using the revised simplex method to solve the master problems. Results show that the D-W algorithm using a 1-5 percent accuracy tolerance gives a more efficient algorithm than the revised simplex method for solving the general Model II problems, and the actual accuracy of the solutions determined is typically within one-half of the imposed tolerance. The problems tested also show that the D-W algorithm is greatly affected by the number of side constraints, while the revised simplex method is greatly affected by the number of initial strata.

Comparison of the computational results using different accuracy tolerances suggests that the D-W algorithm has a slow asymptotic convergence rate in the final iterations, but makes good progress in the initial iterations. The storage requirement for the D-W algorithm is far less than that for the revised simplex method.

The Primal-Dual Conjugate Subgradient Algorithm is coded and tuned to solve general Model II problems. Results show that the computational time of this algorithm is greatly affected by the number of side constraints. For the a-type problems, Algorithm PDCSA can give a more efficient algorithm than the revised simplex method. However, the duality gap is large for most of the test problems. Additional strategies are recommended by Sherali and Ulular (1988) can be applied to reduce this duality gap. Alternatively, one can use a finitely convergent Augmented Lagrangian penalty approach for this problem as described by Bertsekas (1982), for example. In this context, if one has multiple diagonal blocks whose separable structures are advantageous to exploit, as in our case, the newly proposed technique of Ruszczynski (1989) may be used.

For most harvest scheduling problems, there are usually a large number of decision variables and few side constraints. If the optimality tolerance is suitably selected, then the D-W algorithm is superior to the revised simplex method for large-scale problems in terms of both computational time and storage requirement. When the available storage is limited, or when the problem size is very large, and when the number of side constraints are relatively small, and there are multiple species or diagonal blocks in the model, then the D-W algorithm would be significantly better than a standard linear programming package for solving such harvest scheduling problems.

Recently, several other methods for solving large-scale linear programming problems have been developed. Among these methods, those that can take advantage of Algorithm LPA are good candidates for solving large-scale harvest scheduling problems. For example, results from Karmarkar's (1984) projection algorithm, Poliak and Tretiakov's (1974) Largrangian penalty function method, Korpelevich's (1977) extragradient method, and Shang Yi's (1987) saddle surface algorithm show that these methods might provide more efficient algorithms than the revised simplex method. So, future research may investigate the application of these methods to solve large-scale harvest scheduling problems.

BIBLIOGRAPHY

1. Bare, B. B., D. G. Briggs, J. P. Roise, and G. F. Schreuder. 1984. A survey of system analysis models in forestry and the forest products industries. *European J. of Operational Research*, 18:1-18.

2. Bazaraa, M. S. and J. J. Jarvis. 1977. *Linear programming and network flows*. John Wiley & Sons, New York. 565 p.

3. Berck, P. and T. Bible. 1984. Solving and interpreting large-scale harvest scheduling problems by duality and decomposition. *Forest Science*, 30:173-182.

4. Bertsekas, D. P. 1982. *Constrained optimization and Lagrange multiplier methods*. Academic Press, New York, N.Y.

5. Clutter, J. L. 1968. MAX-MILLION-a computerized forest management planning system. School of For. Res., Univ. of Ga. 24 p.

6. Clutter, J. L., J. C. Fortson, L. V. Pienaar, G. H. Brister, and R. L. Bailey. 1983. *Timber management: a quantitative approach*. John Wiley & Sons, New York. 333 p.

7. Dantzig,G. B. and P. Wolfe. 1960. Decomposition principles for linear programs. *Operations Research*, 8:101-111.

8. Dykstra, D. P. 1984. *Mathematical programming for natural resource management*. McGraw-Hill Book Company, New York. 318 p.

9. Hoganson, H. M. and D. W. Rose. 1984. A simulation approach for optimal timber management scheduling. *Forest Science*, 30:220-238.

10. Johnson, K. N. 1977. A comment on techniques for prescribing optimal timber harvest and investment under different objectives - discussion and synthesis. *Forest Science*, 23:444-446.

11. Johnson, K. N. and S. A. Crim. 1986a. FORPLAN version 1: structures and options guide. USDA Forest Service, Fort Collins, CO. 214 p.

12. Johnson, K. N. and S. A. Jones. 1979. A user's guide to multiple use sustained yield resource scheduling calculation (MUSYC). USDA Forest Service, Fort Collins, CO. 243 p.

13. Johnson, K. N. and H. L. Scheurman. 1977. Techniques for prescribing optimal timber harvest and investment under different objectives - discussion and synthesis. *Forest Science Monograph* 18, 31 pp.

14. Johnson, K. N. and T. W. Stuart. 1986b. FORPLAN version 2: mathematical programmer's guide. USDA Forest Service, Fort Collins, CO. 156 p.

15. Iverson, D. C. and R. M. Alston. 1986. The genesis of FORPLAN: a historical and analytical review of Forest Service planning models. USDA Forest Service Gen. Tech. Rep. INT-214. 31 p.

16. Karmarkar, N. 1984. A new polynomial-time algorithm for linear programming. Working paper, AT&T Bell Laboratories, Murray Hill, New Jersey. 38 pp.

17. Korpelevich, G. M. 1977. The extragradient method for finding saddle points and other problems. *Matekon* 13(4):35-49.

18. Lasdon, L. S. 1970. *Optimization theory for large systems.* MacMillan Pub. Co., New York. 523 p.

19. Liittschwager, J. M. and T. H. Tcheng. 1967. Solution of a large scale forest scheduling problem by linear programming decomposition. *J. Forestry,* 65:644-646.

20. Navon, D. I. 1971. Timber RAM ... a long-range planning methods for commercial timber lands under multiple-use management. U.S.D.A. For. Ser., Res. Paper PNW-70. 30 p.

21. Poliak, B.T. and N. V. Tretiakov. 1974. An iterative method for linear programming and its economic interpretation. *Matekon* 10(1):81-100.

22. Ruszczynski, A. 1989. An Augmented Lagragian decomposition method for large-scale linear programming problems. *Operations Research Letters,* forthcoming.

23. Sen, S. and H. D. Sherali. 1986. A class of convergent primal-dual subgradient algorithms for decomposable convex problems. *Mathematical Programming,* 35(3):279-297.

24. Shang Yi. 1987. A new polynomial algorithm for linear programming (saddle surface algorithm). *J. of Shenyang Institute of Chemical Technology.* 22 pp.

25. Sherali, H. D. and O. Ulular. 1988. A primal-dual conjugate subgradient algorithm for specially structured linear and convex programming problems. in *Applied Mathematics and Optimization,* forthcoming.

26. Ware, G. O. and J. L. Clutter. 1971. A mathematical programming system for the management of industrial forest. *Forest Science,* 17:428-445.